Institute of Mathematical Sciences and Technology

Norwegian University of Life Sciences

CFD with OpenSource software, assignment 3

Tutorial Snowdrift modeling, twoPhaseEulerFoam

Author:
Jan Potac

Peer reviewed by:
Håkan Nilsson
Niklas Järvstråt

January 25, 2010

Scope

The goal of this tutorial is to setup and run 2D transient simulation of incompressible two-phase turbulent flow for the purpose of modeling of blowing snow and snow deposition. To do so, the appropriate solver and boundary conditions have to be chosen and defined. First, basics of snow and earth boundary layer physics important for modeling of blowing snow are introduced. The second part covers brief selection of suitable solver in OpenFOAM. Finally, a simple application of particular solver is demonstrated.

1 Theory

1.1 Blowing snow and boundary layer

Snow and boundary layer physics generally covers wide spectra of problems, while only some are important for blowing snow modeling. Aerodynamical properties are the most crucial to satisfy proper snow-air interface behavior. Generally, the formation of snow in atmosphere is complicated process and wide variety of snow particles with different shapes are created. The most well known shapes are classified as snow crystals, flakes and graupels. These primary particles change when transported through the atmosphere. When settled on open areas, the redistribution of snow becomes dominant role in snowdritfing. There exist three modes of transport of blown snow defined as turbulent diffusion, saltation, and creep. These modes are dependent on several factors as wind speed, turbulence intensity, surface roughness, etc. The turbulent diffusion is characterized as vertical transport of snow particles without necessarily touching the ground. The thickness of such layer varies between 10 and 100 m. The thickness of saltation layer, in which particles are ejected from the surface and after traveling curved trajectory under the influence of wind and gravity forces, rebounded or eject other particles, is usually between 10 and 100 cm. During the creep particle migrate along the surface without loosing its contact. The thickness of the creep layer is usually in order of millimeters.
As was mentioned above, the shape of snow particles varies significantly, while the particle physical properties vary as well. This results in wide variety of crystal diameters, density and viscosity. Generally the particle diameter varies between range of micrometers to centimeters, the density between 50 $\frac{kg}{m^3}$ and 500 $\frac{kg}{m^3}$. Note that the density of ice is approximately 900 $\frac{kg}{m^3}$. The difference is mainly caused by air content inside the snow particle. The dynamic viscosity of snow particles is approximately $9e^{-8}\ Pa \cdot s$.
A wind and gravity are the main transport forces for air dispersed snow. Within the earth boundary layer, especially close to the surface, there is a need to involve vertical wind velocity and snow concentration profiles. Based on measurements of wind speed in different heights above the ground, the logarithmic and power law wind profiles have been constructed. The logarithmic law is applied up to 100 m above the ground. The power law relationship is applied when the surface roughness or atmospheric stability information is not available.
The relation describing the logarithmic wind profile under neutral atmospheric stability condition can be expressed as

$$u(z) = \frac{u_*}{\kappa} \ln\left(\frac{z-d}{z_0}\right) \tag{1}$$

where $u(z)$, u_*, κ, d and z_0 is the wind velocity at height z, friction velocity, von Karman's constant (≈ 0.41), zero plane displacement and surface roughness, respectively.

The power law relationship is based on reference wind speed and reference height values and can be written as

$$u(z) = u_r \left(\frac{z}{z_r} \right)^{\alpha} \tag{2}$$

where u_r and z_r is the reference wind speed and height, respectively. The coefficient α characterizes the stability of the atmosphere. For neutral stability condition the α is approximately $\frac{1}{7}$. To obtain the realistic coefficients present in (1) the experimental data taken from two discrete heights, z_1 and z_2, can be used. The surface roughness is then calculated as

$$z_0 = \exp \frac{u(z_2) \ln z_1 - u(z_1) \ln z_2}{u(z_2) - u(z_1)} \tag{3}$$

The friction velocity can be calculated as

$$u_* = \frac{u(z_i)\kappa}{\ln \frac{z_i}{z_0}} \tag{4}$$

The concentration of blowing snow primary depends on climatic conditions inside clouds, and secondary on the distance above a surface. The concentration of snow rises close to the surface due to the transport of already settled but not bounded particles. Based on experiments, the concentration profile for the turbulent diffusion mode can be calculated as

$$c(z) = c_r \left(\frac{z}{z_r} \right)^{-\frac{U_F \sigma_s}{\kappa u_*}} \tag{5}$$

where $c(z)$, c_r, U_F, σ_s is the concentration of snow at height z, reference concentration, particle settling velocity and Schmidt number, respectively. The term $U_F \sigma_s$ can be calculated using experimental relation $U_F \sigma_s = 0.38 u_* + 0.12$.
In multiphase numerical models, the volume fraction is mostly used instead of concentration. It is defined as a ratio of volume of dispersed discrete phase to that of the control volume. The value of snow volume fraction of 0.0001 might be used.
To depict the snow surface, the distribution of volume fraction might be used. There exist also another method using mesh deformation. In this approach the actual friction velocity is calculated and compared to some threshold value. Based on this the surface erosion or snow deposition is modeled.

1.2 Multiphase modeling in OpenFOAM

OpenFOAM natively provides several multiphase solvers:

- bubbleFoam
- cavityFoam
- interFoam group
- settlingFoam
- twoLiquidMixingFoam
- twoPhaseEulerFoam

There exist several solvers classified as interFoam group, which are built based on Volume of Fluid method. This method was developed to simulate free surface for fluid-fluid interface problems. In this method, volume fraction is defined as a discontinuous function altering between values 0 and 1, and thus is not able to cover problems with non-sharp interface.
For a purpose of dispersion problems the solvers as bubbleFoam, settlingFoam, twoPhaseEulerFoam have been developed. In the OpenFoam documentation these solvers are characterized as solvers for a system of 2 incompressible fluid phases with one phase dispersed, e.g. gas bubbles in a liquid.
The twoPhaseEulerFoam solver, compare to the other ones, provides more advanced options as interfacial and kinetic theory models. These models are able to take into account the behavior of as airborne particles as the particles already settled. While the interactions between lifted particles can be neglected, when settled the particle friction becomes dominant due to the rapid increase in amount of neighboring particles.
For basic solver description see twoPhaseEulerFoam tutorial by Praveen Prabhu Baila from 2007.

2 Test case

A 2D test case for transient solution of two-phase turbulent flow is characterized by a dispersed solid-fluid mixture entering a domain, passing an obstacle and leaving immediately. The inlet boundary condition for logarithmic wind profile is used according to (1). The uniform inlet snow volume fraction is to the value of 0.0001. The surfaces representing ground and obstacle are considered as no-slip walls. The transport properties are set based on appropriate physical values.

2.1 Geometry

The simple geometry contains a wind tunnel with a cubic obstacle placed on the bottom. The tunnel is 10 m high and 56 m long. The cube has dimensions of 1x1 m and is located 25 m downstream. See Fig. 1. All the domain surfaces used in boundary condition definition and setup are called as follows: inlet, outlet, obstacle, sky and frontAndBack.
First, download and untar the attached case files into your home run directory. Open the blockMeshDict file and check its structure. When finished create the mesh.

```
cd snowDrift
vim constant/polyMesh/blockMeshDict
blockMesh
checkMesh
```

The generated mesh and geometry can be seen at Figure 1.

3 Initial and boundary conditions

The next step covers setup of initial and boundary conditions (BCs). These are located in subdirectory `0`. The files `Ua`, `Ub`, `p`, `k`,`epsilon`, `alpha` and `Theta` have been edited to fulfill boundary conditions introduced above. First, in both files, `Ua` and `Ub`, the inlet logarithmic wind profile, which compilation is described later on, and ground and obstacle no-slip wall conditions are defined. The no-slip wall BC is defined using `partialSlip` of a `valueFraction` of 0. Note that the value of 1 corresponds to slip.

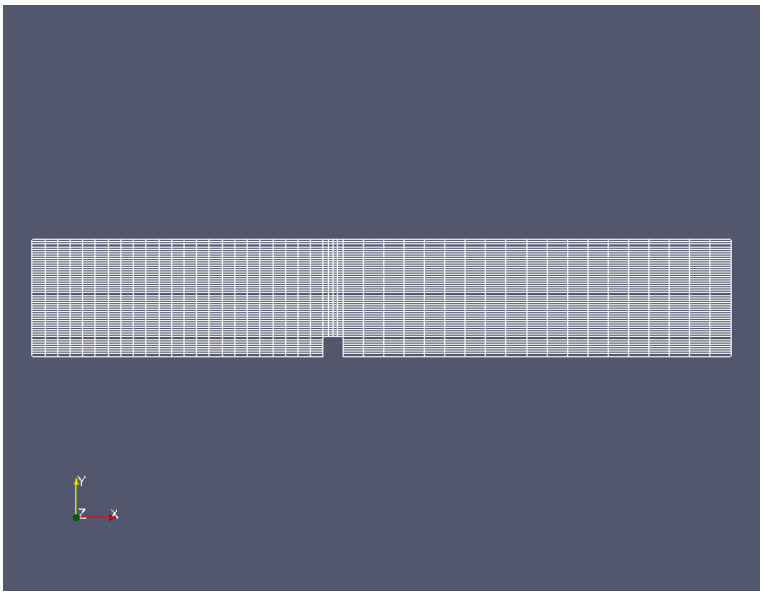

Figure 1: Geometry and mesh

```
inlet
    {
        type                    logarithmicVelocity;
        n                       (1 0 0);
        y                       (0 1 0);
        Vel1                    5.5;
        Vel2                    8;
        h1                      1;
        h2                      2;
        value                   uniform (0 0 0);

ground
    {
        type            partialSlip;
        valueFraction   uniform 0;
    }

obstacle
    {
        type            partialSlip;
        valueFraction   uniform 0;
    }
```

The pressure BC, file `p`, follows the zeroGradient condition on walls and at the outlet, while the inlet is specified by `fixedValue` of 0.

```
 inlet
    {
```

```
        type            fixedValue;
        value           uniform 0;
    }

 outlet
    {
        type            zeroGradient;
    }
```

The files for turbulence kinetic energy, `k`, and energy dissipation, `epsilon`, are defined using `fixedValue` at the inlet, `inletOutlet` at the outlet, and `zeroGradient` at the ground, obstacle and sky.
The granular temperature `Theta` is defined as `zeroGradient` except inlet and outlet where the the `fixedValue` is specified.
The last BC to set is the volume fraction, `alpha`, using `fixedValue` at the inlet, `inletOutlet` at the outlet, otherwise `zeroGradient`.

```
 inlet
    {
        type              fixedValue;
        value             uniform 0.0001;
    }

 outlet
    {
        type              inletOutlet;
        inletValue        uniform 0.0001;
```

3.1 Inlet velocity profile

Before the logarithmic wind profile inlet BC can be used, it has to be implemented into the solver. The implementation of logarithmic function is found Member Function. Before the function is specified itself, related variables have to be introduced in headers and class templates. For better understanding see both `logarithmicVelocityFvPatchVectorField.H` and `logarithmicVelocityFvPatchVectorField.C` files. The logarithmic function is implemented in Member Function. First, the bound box is specified. Then the vector variable `ctr` which expresses the position the zero level is defined. The vector field `c` is used to point cell centers of a patch. The scalar field `coord` is the relative coordinate varying between values 0 and 1. To obtain the real range of coordinate the `coord` has to be scaled by the appropriate dimension of the boundary.

```
 // * * * * * * * * * * * * * * * * Member Functions  * * * * * * * * * * * * * * * //

void logarithmicVelocityFvPatchVectorField::updateCoeffs()
{
    if (updated())
    {
        return;
    }
```

```
    // Get range and orientation
    boundBox bb(patch().patch().localPoints(), true);
    vector ctr = (bb.min());
    const vectorField& c = patch().Cf();
    // Calculate local 1-D coordinate for the logarithmic profile
    scalar z0 = exp(((Vel2_*log(h1_))-(Vel1_*log(h2_)))/(Vel2_-Vel1_));
    //surface roughness
    scalar k = 0.41; //von Karman constant
    scalar ufr = (Vel1_*k)/(log(h1_/z0)); //friction velocity
    scalarField coord =((c - ctr) & y_)/((bb.max() - bb.min()) & y_);
    vectorField::operator=(n_*(ufr/k)*log(((coord*((bb.max() - bb.min()) & y_))+z0)/z0));
}
```

The Member Function can be easily edited to implement Power Law velocity profile as well by adding the following line after logarithmic function definition.

```
vectorField::operator=n_*Vel1_*pow(coord*((bb.max() - bb.min()) & y_)/h1_,0.143);
//power law velocity profile
```

The velocity profile, which is not preferred or desired is easily inactivated by commenting out the corresponding line.

4 Transport properties and run

In this test case, the density, viscosity and particle diameter is specified based on real physical values corresponding to snow and air. The file `transportProperties` specifying both phases is then

```
phasea
//snow
{
    rho             rho [ 1 -3 0 0 0 ] 250;
    nu              nu [ 0 2 -1 0 0 ] 3.6e-10;
    d               d [ 0 1 0 0 0 0 0 ] 0.0002;
}

phaseb
//air
{
    rho             rho [ 1 -3 0 0 0 ] 1.34;
    nu              nu [ 0 2 -1 0 0 ] 1.328e-05;
    d               d [ 0 1 0 0 0 0 0 ] 1;
}
```

Finally, the case is ready to run using `snowTwoPhaseEulerFoam`. Before that, one must compile the velocity profile. First run `wclean` and `wmake` in snowTwoPhaseEulerFoam directory.

5 Results

First, check the proper velocity inlet profile. Running `paraFoam` and applying the `Cell Center` filter and `Glyph` for inlet patch the both velocity profiles are seen in Figures 2 and 3.

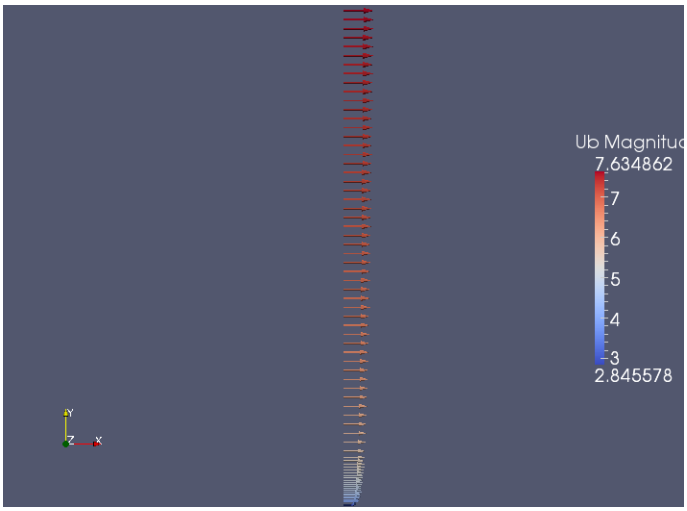

Figure 2: Power law velocity inlet profile

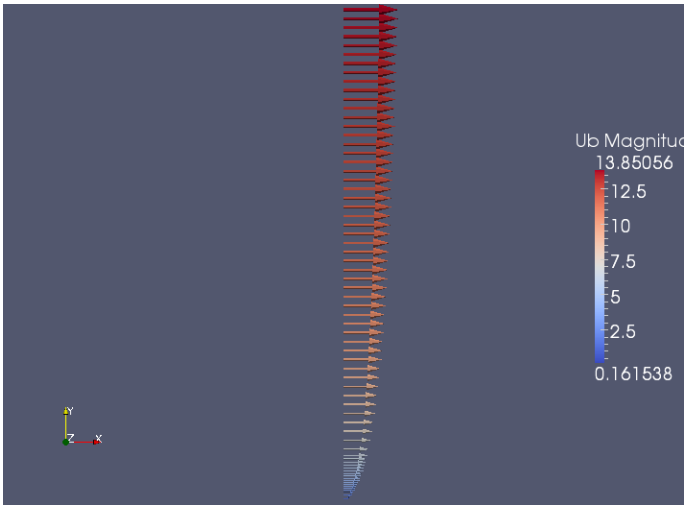

Figure 3: Logarithmic velocity inlet profile

The snow surface is captured by plotting volume fraction, alpha. When take a look at some early time steps, the increase of snow volume fraction on ground boundary is significant. There also appear two regions, one on the windward and leeward sides where the snow concentration is higher compare to inlet value. See Figure 4. The problems occur when the recirculation zone behind the obstacle is formed. There exist two regions oserved on the ground downwind. First one, is further downwind and the other in close vicinity of the obstacle. While the value of alpha inside the first cell on the boundary reaches some

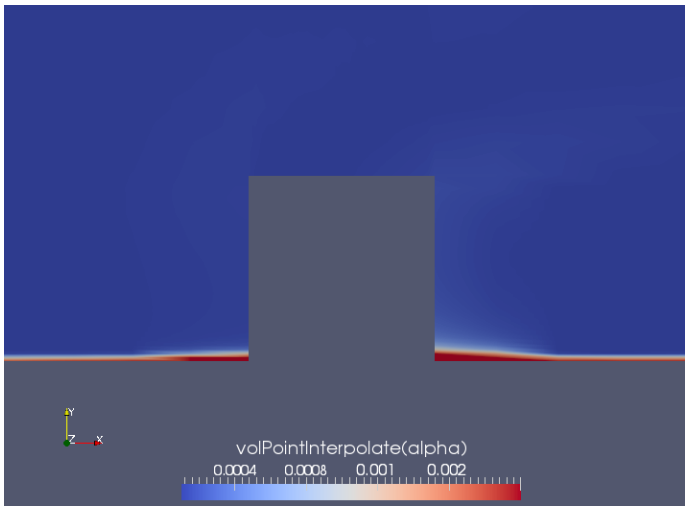

Figure 4: Volume fraction, t = 41

equilibrium value immediately, in the obstacle vicinity this value rises up and then drops down to some equilibrium. This behavior seems be similar to some stream. It means when flow initiated, the particles are carried and pushed by the air to the obstacle. When the recirculation zone formed, the accumulated snow particles start flowing away to reach the free surface. This is also observed by checking the enclosed cells. There is no increase in volume fraction in second layer of cells. See Figure 5. This behavior is probably commited

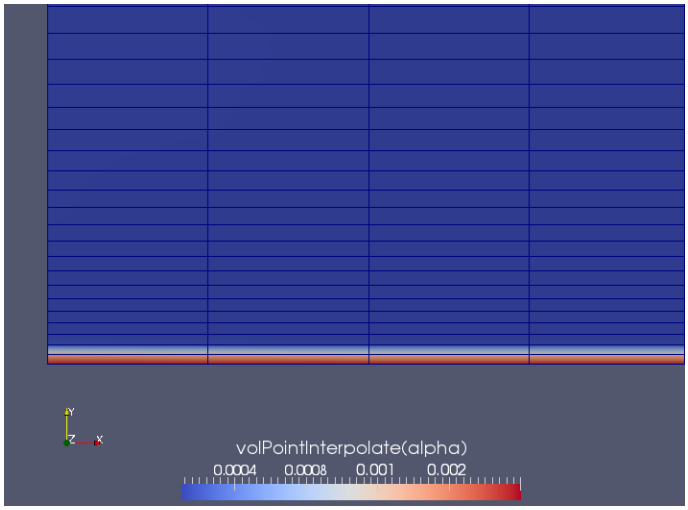

Figure 5: Volume fraction, t = 246

by incorrectly defined friction between particles which allows them to move when already settled. The friction model is a part of kineticTheory model and need to be deeply treated to be relevant for blowing snow modeling.
Rather than ploting an isosurface of volume fraction to represent a real shape of a snow drift, the mesh deformation approach seems to be more relevant due to topology change of the drifted surface.
The rate of snow accumulation depends on snow flux and snow concentration. To model the

snow drifting properly, it is necessary to construct vertical volume fraction profile expressed by (5).